UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/091,645	03/05/2002	Handong Wu	NETAP019	9146

28875 7590 06/23/2006

Zilka-Kotab, PC
P.O. BOX 721120
SAN JOSE, CA 95172-1120

EXAMINER	
HOMAYOUNMEHR, FARID	

ART UNIT	PAPER NUMBER
2132	

DATE MAILED: 06/23/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/091,645
Filing Date: March 05, 2002
Appellant(s): WU ET AL.

Kevin J. Zilka
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed April 12, 2006 appealing from the Office action mailed November 30, 2005.

(1) Real Party in Interest

The statement identifying the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

Following documents were relied upon in rejection of claims:

6,279,113	Vaidya	6-1998
2003/0101358	Porras	11-2001

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 112

9.1. Claims 21 and 22 are rejected under 35 U.S.C. 112, first paragraph, because the specification, while being enabling for general functionality of the APIs, does not reasonably provide enablement for the particular APIs cited in the claims. The specification does not enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make the invention commensurate in scope with these claims.

9.1.1. In claim 21 the API is claimed to have the form of frame_context_pointer_position. What is the exact implication of this form? Does it identify the parameters exchanged between objects? How is an API that is compliant with this form distinguished from the one that is not? What is the specific advantage of this form?

9.1.2. In claim 22, the API is requested to include at least one of frame_tcp_bridge, frame_udp_bridge, frame_ip_bridge, frame_http_bridge. Page 12 of the specification merely mentions the incision of these items as an

example. No description on the specific functionality or how to use these elements is provided.

9.2. Claims 21 and 22 are rejected as failing to define the invention in the manner required by 35 U.S.C. 112, second paragraph.

The claim(s) are narrative in form and replete with indefinite and functional or operational language. The structure which goes to make up the device must be clearly and positively specified. The structure must be organized and correlated in such a manner as to present a complete operative device. The claim(s) must be in one sentence form only. Note the format of the claims in the patent(s) cited.

Claim Rejections - 35 USC § 103

9.3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9.4. Claims 1 to 19 are rejected under 35 U.S.C. 102(e) as anticipated by or, in the alternative, under 35 U.S.C. 103(a) as obvious over Vaidya (US Patent No. 6,279.113) and further in view of Porras (US Patent Application Publication number 2003/0101358 A1, filed 11/28/2001).

9.4.1. As per claims 1, Vaidya is directed to an intrusion detection and analysis system (Fig.1 column 5 lines 5 to 9) comprising:
a data monitoring device comprising a capture engine operable to capture data passing through the network (Fig. 1, item 10 column 5, lines 9 to 25) in response to a trigger (as per Fig. 3, the data monitor 58 continues to monitor data when triggered by item 64. see also column 7 line 4 to 6) and configured to monitor network traffic (Fig. 1, item 10 column 5, lines 13 to 15, see also Fig. 4 item 36 column 7 lines 11 to 15), decode protocols (Fig. 2, item 36 within item 10, column 7 lines 17 to 24), for grouping packets into different protocol presentations and assembling the packets into high level protocol groups (column 7 line 16 to 24 indicates that Vaidya determines the layer of OSI stack to which the monitored packet belongs and performs intrusion detection based on profiles associated with OSI layers), and analyze received data (column 6 line 57 to column 7 line10) for managing the network by collecting statistics (column 8 lines 15 to 40 and column 9 lines 4 to 21 in conjunction with the response to claim 9 in the first office action show collection of statistics by Vaidya), and detecting broken lines,

traffic loads, and network errors (as described in column 3 line 11 to 26, the attack signature profiles used by Vaidya's system to perform intrusion detection include profiles to detect data transport alteration or deletion, and malfunction of network objects. Broken lines cause deletion of transport data and malfunction of network objects causes network errors. Traffic load is measured by obtaining statistics on data packets traveling in the network. Hence Vaidya's system is capable of collecting statistics and detecting broken lines, traffic loads and network errors), and;

an intrusion detection device (Fig 2, item 36) separate from the data monitoring device (Vaidya's system obviously discloses the functionally separate data monitoring device and intrusion detection device. See section 8 above), the intrusion detection device comprising a detection engine operable to perform intrusion detection on data provided by the data monitoring device (item 36 column 6 lines 7 to 13);

application program interfaces configured to allow the intrusion detection device access to applications of the data monitoring device to perform intrusion detection. Application program interface is defined as 'a set of functions or methods used to access some functionality'. Vaidya does not explicitly mention the use of APIs, however, referring to Fig. 2, the configuration builder module (item 32) allows the intrusion detection device (item 36) access attack signature

profiles stored in signature profile memory (item 39), as described in column 6 lines 1 to 11. Also, the communication module (item 34) allows intrusion detection device access the data in database handler (item 26). This clearly allows the intrusion detection device access the functionality of the data-monitoring device to perform intrusion detection, and hence discloses the feature. Therefore, the Examiner asserts that Vaidya discloses the feature by inheritance.

Vaidya does not specifically refer to an application program interface to access the data and functionalities of the data-monitoring device, but Porras clearly suggests the use of APIs (paragraphs 40 to 42 and claim 2) in development of Intrusion Detection Systems. It would have been obvious to a person skilled in art to use APIs, as clearly disclosed by Porras, as a means to transfer information between objects or elements of a distributed system in order to build the intrusion detection system invented by Viadya. The motivation to do so would have been to take advantage of an already prepared and well tested element to perform part of the required functionality (transfer of extracted packet information from Vaidya's item 40 to perform intrusion detection, as described in column 7 line 25 to 30).

memory for storing reference network information used by the intrusion detection

device to determine if an intrusion has occurred (Fig 2, item 26, column 5 lines 46 to 50);

wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, by allowing the intrusion detection device to call an application programming interface configured to open protocol decoding application associated with the separate data monitoring device and by allowing the intrusion detection device to call and API configured to open an alarm generation application associated with the separate data monitoring device (as mentioned above Vaidya discloses APIs, by the way of a set of functions or methods to access some functionality. Vaidya discloses the functionality of protocol decoding as described above and in column 7 line 16 to 24. Vaidya also discloses the functionality of alarm generation, as described in rejection of claim 13 in the first office action. Therefore, Vaidya discloses the entire feature).

9.4.2. As per claim 2, Vaidya continues to teach a reference network information comprises a signature database including signature profiles associated with a known network security violation (Fig 2, item 39 column 6 lines 1 to 7) and wherein the detection engine (item 32) is operable to compare the data provided by the data monitoring device (item 39) with the signature profiles to detect network intrusions (column 6 lines 7 to 18).

9.4.3. As per claim 3, Vaidya is directed to a parser (Fig. 4 item 36) operable to parse (referring to Fig. 8 and column 9 lines 45 to 55, and Fig. 9 and column 10 line 17 to column 11 line 15, the data collector, as part of processing the attack signature profile, separates the fields in the signature profile, which is equivalent to parsing), generate (Fig. 2 and 3 and column 6 lines 27 to 29), and load (Fig. 2 column 6 lines 1 to 11 and Fig. 4 column 7 lines 18 to 23) signatures at the detection engine.

9.4.4. As per claim 4, Vaidya is directed to the reference network information comprises a baseline state of network traffic (state cache as shown in Fig. 4 item 44 and described in column 9 lines 3 to 20 stores the network traffic state data) and wherein the detect engine is operable to compare the data received by the capture engine to the baseline network state and look for anomalies (claims 11 and 12).

9.4.5. As per claim 5, Vaidya is directed to a data-monitoring device in accordance with claim 4 that provides the baseline state of network traffic. Vaidya provides the baseline state of network traffic in the virtual processor (Fig. 4 item 36), wherein the Register Cache stores the information extracted from a packet (as described in column 7 lines 11 to 17), and in the case where a sequential or timer/counter based signature profiles are invoked (column 7 lines 48 to 51), the state of network traffic (data extracted from the packet) is saved in

the state cache (Fig 4. item 44) and used in conjunction with the data from the subsequent packet (column 7 lines 59 to 65) to determine network anomalies. The use of traffic state data to detect network anomalies is further described in column 8 lines 16 to 40.

9.4.6. As per claim 6, Vaidya is directed to a log file configured to at least temporarily store reports generated by the detect engine, on account of the Reaction Module (Fig. 2 item 38), which receives alerts from the detection device whenever a network intrusions is detected, and initiates reactions depending on nature of the attack (column 6 lines 18 to 26).

9.4.7. As per claim 7, Vaidya is directed to a system according to claim 6, further comprising an alarm manager operable to generate alarms based on the information in the log file (column 6 lines 21 to 26).

9.4.8. As per claim 8, Vaidya discloses a system in accordance with claim 1 further comprising a filter configured to filter out packets received at the data-monitoring device. Filter is defined as 'a device that separates data in accordance with specific criteria'. Item 96 in Fig. 6 (which describes the operation of the data collector) returns 'No Entry Found' when the data collected from the packet indicates that the packet's destination server is not being monitored, and no further action will take place on that packet (column 8 lines 58

to 65). This clearly discloses 'a separation of packets based on a criteria' and hence discloses a 'filter'. Vaidya does not specifically refer to a filter. Therefore, the Examiner asserts that it Vaidya discloses the feature.

9.4.9. Claim 9 has been cancelled by the applicant.

9.4.10. As per claim 10, Vaidya is directed to a system in accordance with claim 1 wherein the capture engine is configured to forward packets and temporarily store packets for later analysis by data monitoring device (Fig. 4 item 40 column 7 lines 15 to 24).

9.4.11. As per claim 11, Vaidya is directed to a method for performing intrusion detection with an intrusion detection and analysis system (Fig.1 and column 5, as described in lines 5 to 9) comprising a data monitoring device including a capture engine operable to capture data passing through the network in response to a trigger (as per Fig. 3, the data monitor 58 continues to monitor data when triggered by item 64. see also column 7 line 4 to 6) and configured to monitor network traffic (Fig. 1, item 10 column 5, lines 13 to 15, see also Fig. 4 item 36 column 7 lines 11 to 15), decode protocols (Fig. 2, item 36 within item 10, column 7 lines 17 to 24), for grouping packets into different protocol presentations and assembling the packets into different protocol presentations and assembling the packets into high level protocol groups (column 7 line 16 to

24 indicates that Vaidya determines the layer of OSI stack to which the monitored packet belongs and performs intrusion detection based on profiles associated with OSI layers), and analyze received data (column 6 line 57 to column 7 line10) for managing the network by collecting statistics (column 8 lines 15 to 40 and column 9 lines 4 to 21 in conjunction with the response to claim 9 in the first office action show collection of statistics by Vaidya), and detecting broken lines, traffic loads, and network errors (as described in column 3 line 11 to 26, the attack signature profiles used by Vaidya's system to perform intrusion detection include profiles to detect data transport alteration or deletion, and malfunction of network objects. Broken lines cause deletion of transport data and malfunction of network objects causes network errors. Traffic load is measured by obtaining statistics on data packets traveling in the network. Hence Vaidya's system is capable of collecting statistics and detecting broken lines, traffic loads and network errors), and an intrusion detection device (Fig 2, item 36) separate from the data monitoring device (Vaidya's system obviously discloses the functionally separate data monitoring device and intrusion detection device. See section 8 above), coupled to data monitoring device (Fig. 2 items 34 and 30 couple the intrusion detection device to the data collected by the data monitoring device) and configured to perform intrusion detection on data provided by the data monitoring device (column 6 lines 1 to 21); the method comprising:

receiving data at data monitoring device (Fig 4. item 36 column 7 lines 18 to 24);

capturing at least a portion of the packets contained within the data (column 7 lines 18 to 24);

by allowing the intrusion detection device to call at least one application program interface configured to open application of the data monitoring device; and performing intrusion detection at the intrusion detection device utilizing at least one of the applications of the data monitoring device (Application program interface is defined as 'a set of functions or methods used to access some functionality'. Referring to Fig. 2, the configuration builder module (item 32) allows the intrusion detection device (item 36) access attack signature profiles stored in signature profile memory (item 39), as described in column 6 lines 1 to 11. Also, the communication module (item 34) allows intrusion detection device access the data in database handler (item 26). This clearly allows the intrusion detection device access the functionality of the data-monitoring device to perform intrusion detection, and hence discloses the feature.

Vaidya does not specifically refer to an application program interface to access the data and functionalities of the data-monitoring device, but Porras clearly suggests the use of APIs (paragraphs 40 to 42 and claim 2) in development of Intrusion Detection Systems. It would have been obvious to a person skilled in art to use APIs, as clearly disclosed by Porras, as a means to transfer

information between objects or elements of a distributed system in order to build the intrusion detection system invented by Viadya. The motivation to do so would have been to take advantage of an already prepared and well tested element to perform part of the required functionality (transfer of extracted packet information from Vaidya's item 40 to perform intrusion detection, as described in column 7 line 25 to 30).

wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, by allowing the intrusion detection device to call an application programming interface configured to open protocol decoding application associated with the separate data monitoring device and by allowing the intrusion detection device to call and API configured to open an alarm generation application associated with the separate data monitoring device (as mentioned above Vaidya discloses APIs, by the way of a set of functions or methods to access some functionality. Vaidya discloses the functionality of protocol decoding as described above and in column 7 line 16 to 24. Vaidya also discloses the functionality of alarm generation, as described in rejection of claim 13 in the first office action. Therefore, Vaidya discloses the entire feature).

9.4.12. Claim 12 is cancelled by the applicant.

9.4.13. Claim 13 is cancelled by the applicant.

9.4.14. As per claim 14, Vaidya discloses a method of claim 11 further comprising filtering prior to capturing packets. Filtering is defined as 'a method to separate data in accordance with specific criteria'. Item 96 in Fig. 6 (which describes the operation of the data collector) returns 'No Entry Found' when the data collected from the packet indicates that the packet's destination server is not being monitored, and no further action will take place on that packet (column 8 lines 58 to 65). This clearly discloses 'a separation of packets based on a criteria' and hence discloses 'filtering'. Vaidya does not specifically refer to filtering. Therefore, the Examiner asserts that Vaidya discloses the feature.

9.4.15. As per claim 15, Vaidya discloses a method of claim 11 wherein performing intrusion detection comprises performing signature matching (Fig. 4 column 7 line 31 to column 8 line 40 and claim 1).

9.4.16. As per claim 16, Vaidya discloses a method of claim 15 wherein the application program interfaces provide parsing of signatures used in signature matching (column 10 lines 17 to 45).

9.4.17. Claim 17 is cancelled by the applicant.

9.4.18. As per claim 18, Vaidya discloses a method of claim 11 wherein performing intrusion detection comprises detecting anomalies in the received data (column 6 line 57 to column 7 line10).

9.4.19. As per claim 19, Vaidya discloses a computer program product (in form of a set of instructions to be sequentially executed as described in column 16 claim 18) for performing intrusion detection with an intrusion detection and analysis system comprising a data monitoring device including a capture engine operable to capture data passing through the network in response to a trigger (as per Fig. 3, the data monitor 58 continues to monitor data when triggered by item 64. see also column 7 line 4 to 6) and configured to monitor network traffic (Fig. 1 column 5 lines 5 to 9) comprising a data monitoring device configured to monitor network traffic (Fig. 1, item 10 column 5, lines 13 to 15, see also Fig. 4 item 36 column 7 lines 11 to 15), decode protocols (Fig. 2, item 36 within item 10, column 7 lines 17 to 24) for grouping packets into different protocol presentations and assembling the packets into different protocol presentations and assembling the packets into high level protocol groups (column 7 line 16 to 24 indicates that Vaidya determines the layer of OSI stack to which the monitored packet belongs and performs intrusion detection based on profiles associated with OSI layers), and analyze received data (column 6 line 57 to column 7 line10) for managing the network by collecting statistics (column 8 lines 15 to 40 and column 9 lines 4 to 21 in conjunction with the response to claim 9 in the first office action show

collection of statistics by Vaidya), and detecting broken lines, traffic loads, and network errors (as described in column 3 line 11 to 26, the attack signature profiles used by Vaidya's system to perform intrusion detection include profiles to detect data transport alteration or deletion, and malfunction of network objects. Broken lines cause deletion of transport data and malfunction of network objects causes network errors. Traffic load is measured by obtaining statistics on data packets traveling in the network. Hence Vaidya's system is capable of collecting statistics and detecting broken lines, traffic loads and network errors), and an intrusion detection device (Fig 2, item 36) separate from the data monitoring device (Vaidya's system obviously discloses the functionally separate data monitoring device and intrusion detection device. See section 8 above), coupled to data monitoring device (Fig. 2 items 34 and 30 couple the intrusion detection device to the data collected by the data monitoring device) and configured to perform intrusion detection on data provided by the data monitoring device (column 6 lines 1 to 21); the product comprising:

code that receives data at data monitoring device (Fig 4. item 36 column 7 lines 18 to 24);

code that captures at least a portion of the packets contained within the data (column 7 lines 18 to 24);

code that calls at least one application program interface configured to open application of the data monitoring device; and performs intrusion detection at the intrusion detection device utilizing at least one of the applications of the data monitoring device. (Application program interface is defined as 'a set of functions or methods used to access some functionality'. Referring to Fig. 2, the configuration builder module (item 32) allows the intrusion detection device (item 36) access attack signature profiles stored in signature profile memory (item 39), as described in column 6 lines 1 to 11. Also, the communication module (item 34) allows intrusion detection device access the data in database handler (item 26). This clearly allows the intrusion detection device access the functionality of the data monitoring device to perform intrusion detection, and hence discloses the feature. (Vaidya does not specifically refer to an application program interface to access the data and functionalities of the data-monitoring device, but Porras clearly suggests the use of APIs (paragraphs 40 to 42 and claim 2) in development of Intrusion Detection Systems. It would have been obvious to a person skilled in art to use APIs, as clearly disclosed by Porras, as a means to transfer information between objects or elements of a distributed system in order to build the intrusion detection system invented by Viadya. The motivation to do so would have been to take advantage of an already prepared and well tested element to perform part of the required functionality (transfer of extracted packet information from Vaidya's item 40 to perform intrusion detection, as described in column 7 line 25 to 30)

; and

a computer-readable storage medium for storing codes (Fig. 2 item 39 and associated description).

wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, by allowing the intrusion detection device to call an application programming interface configured to open protocol decoding application associated with the separate data monitoring device and by allowing the intrusion detection device to call and API configured to open an alarm generation application associated with the separate data monitoring device (as mentioned above Vaidya discloses APIs, by the way of a set of functions or methods to access some functionality. Vaidya discloses the functionality of protocol decoding as described above and in column 7 line 16 to 24. Vaidya also discloses the functionality of alarm generation, as described in rejection of claim 13 in the first office action. Therefore, Vaidya discloses the entire feature).

9.4.20. Claim 20 is rejected because it depends on claim 19 which is rejected.

(10) Response to Argument

Appellant arguments are categorized in several groups. Examiner's response to each group is as follows:

Response to Issue # 1:

In response to rejection under 35 U.S.C. 112, first paragraph of claims 21 and 22, appellant refers to Page 12 of disclosure. Appellant admits the page provides merely the form the API. However, simply defining a form for an API is not sufficient to enable one skilled in art to make such API. In the previous office actions, Examiner has requested enablement for a person skilled in art to make such API. Particularly in the office action dated 11/30/2005, section 9.2, the exact functionality of API was requested. The specification does not provide the feature requirements of particular API's, their exact functionality, an example of a code, or anything that enables a person skilled in art to make the API. Defining a form of an API, as it appears in Page 12 of specifications, is not enabling, as it does not disclose what functionality is being communicated on requests between the intrusion detection device and the data monitoring device.

Response to Issue # 2:

In response to rejection under 35 U.S.C. 112, second paragraph of claims 21 and 22, appellant simply finds the rejection baseless. However, appellant's definition of the

API's is limited to a form. The functionality of API's such as frame_context_pointer_position, frame_tcp_bridge, frame_udp_bridge, frame_ip_bridge, frame_http_bridge is not defined anywhere in the disclosure. Claims are indefinite based in the lack of enablement expressed above.

Response to Issue # 3:

In response to rejections under 35 U.S.C. 102(e) as anticipated by or, in the alternative, under 35 U.S.C. 103(a) as obvious over Vaidya (US Patent No. 6,279.113) and further in view of Porras (US Patent Application Publication number 2003/0101358 A1, filed 11/28/2001), the applicant has argued the following:

Group #1: Claims 1-8, 10, 14-15, and 18-20

Appellant has argued that Vaidya does not disclose the claims because the claims require an intrusion detection device **separate** from the data monitoring device. Examiner's response dated 11/30/2005, indicated the only separation implied in the claims is a functional separation, and Vaidya discloses the claims in a system including functionally separate devices. Applicant has argued, by directing to page 7, line 14-page 8, line 6 and Figure 1 that the network analysis and intrusion detection devices are separate devices. However, Figure 1 is a functional block diagram and hence the separate devices imply only functional separation. Furthermore, Figure 1 indicates a

device 18, which is comprised of two devices. Appellant's element 18 is shown in Figure 2 as one item called NIDAS in the network. Therefore, the separation described in disclosure implies nothing more than a functional separation, and the broad interpretation of claims and figures is that the system is within one element (NIDAS, device 18 of Figure 2). In addition, functional separation is not excluded in the claim language. Meaning, even if physical, architectural or other separations can be interpreted, a functional separation is also a valid interpretation. Therefore, a rejection based on functional separation is valid.

Appellant has also argued Examiner's assertion that Vaidya is not limited to one processor. Appellant agrees that the reference (Figure 4 of Vaidya) discloses modules that work with the virtual processor, but argues that such modules are not separate processors. Appellant's emphasis is on the word "processors", however, this word does not exist in the claims at hand. Examiner has considered the general meaning of the word "processors", which in the context of the subject at hand, is an element capable of processing data and performing operations. Clearly, items 36, 34 and 38 in Figure 4 are capable of processing data and they do work with the virtual processor. Appellant has further argued that the at the mentioned modules do not provide the separate functionality claimed by the applicant. However, examiner's purpose for referencing items 36, 34 and 38 in the second office action was to indicate that Vaidya is not limited to one processor, and discloses use of separate processors performing different parts of operation. Vaidya's disclosure of appellant claimed functionalities is described in details

in section 9.4. above, which was communicated to the appellant as part of the office first and second actions.

Appellant argues still yet, that the functionality of items 36 and 40, as relied by the examiner, does not meet applicant's specific claim language. However, appellant's argument excludes parts of item 36 other than item 40. In fact, appellant merely argues that item 40 is incapable of performing some functionalities and concludes that items 36 and 40 cannot meet claim language. Item 40 in not the only element relied on by the examiner to disclose claimed functionalities. Once again, section 9.4 above describes how Vaidya fully discloses claimed functionalities. For the purpose of clarity, part of Examiner's office action dated 11/30/2005, which pertains to item 40, shown as a separate part of item 36, is presented as follows: "Referring to figure 4 again, Vaidya performs the functionality of applicant's data monitoring device and intrusion detection device in item 36, however, within item 36 it discloses item 40 as a separate device. Based on Vaidya column 8 line 40 to 55, packet information is extracted and stored in cache register 40. All data packet information, including MAC, IP, Transport and Application layer data is extracted and entered in cache register 40. As this action is explained in Fig. 5, it clearly shows a separate function of packet data extraction and collection and storage. In the following steps, depicted in separate procedures and figures 6 to 10 by Vaidya, the data collected from packets (stored in item 40) is used to perform intrusion detection. Vaidya clearly separates the functionality of data collection, as described in Fig 5, from intrusion detection, as described in Figures 6 to 10.".

Therefore, the purpose of Examiner's assertion that item 40 is within item 36, and shown as a separate entity is to point out another instance of Vaidya's disclosure of using separate devices to perform separate functionalities.

Furthermore, appellant argues that mere claiming separate steps does not meet appellant separate devices, as claimed. Once again, the purpose of Examiner's assertion that Vaidya performs operations in separate steps was to establish that Vaidya inherently discloses separate modules performing separate functionalities. Examiner has not merely relied on separate steps to meet appellant separate devices. The functionality of appellant's claimed devices and how Vaidya meets them is described in section 9.4.

Examiner respectfully asserts that examiner's response made in office action dated 11/30/2005 fully responses to all arguments made in appellant's arguments in the Office Action dated 10/12/2005, and clearly shows how Vaidya completely meets all claimed requirements.

With respect to appellant's claimed technique of using application programming interfaces, appellant suggests that the Examiner has argued that the claim languages does not point out any advantage in separation of intrusion detection device and data monitoring device. This is false, as the pertinent part of Examiner response in section 8.3 of the Final rejection is as follows: "The second basis of applicant's argument is that

the use of API's to call different applications of data monitoring device from the intrusion detection device is not disclosed by Vaidya. In fact, applicant's specifications or claims (original or amended) do not point out any advantage in separation of the intrusion detection device and data monitoring device that is distinct from teaching of Vaidya, except for the use APIs to invoke certain functionalities.". Therefore, examiner did notice the use of API's to call functions in separate modules. In the same office action, examiner clearly details how Vaidya discloses the use of API's as follows: "However, using APIs as a means to transfer information between objects or elements of a distributed system is obvious and well known to a person skilled in the art and it would have been obvious to a person skilled in the art to use the APIs in order to build the intrusion detection system invented by Viadya. The motivation to do so would have been to take advantage of an already prepared and well tested element to perform part of the required functionality (transfer of extracted packet information from Vaidya's item 40 to perform intrusion detection, as described in column 7 line 25 to 30). Even though APIs have been widely know application development tools since before the time of claimed invention, the examiner has cited a reference to reaffirm the general use of APIs in development of Intrusion Detection systems (see Claim Rejections – 35 USC 103)." Therefore, Examiner has established that Vaidya's anticipation of application programming interfaces as one of inherency, because Vaidya's system does have separate modules that exchange information among one another. This requires the Vaidya's system to have a means to call functions in a separate module, which is the general purpose and use of API's. Examiner also asserts that use of API's was well-

known and widely used at the time of invention. Columns 1 and 2 of Vaidya indicates general use of client-server architectures at the time of invention. Client systems call programs and functions on the servers, which is again the essence of Application Programming Interfaces. In addition, and for the purpose of absolute clarity about use of Application Programming Interfaces in the field of Intrusion detection systems, Examiner has cited Porras and used it in combination with Vaidya in a 103 rejection, the merits of which are detailed in section 9.4 above.

Thus, none of the independent claims are allowable.

Group #2: Claim 11

Appellant incorporates the arguments relative to group 1, and again refers to the claim requirement of application programming interfaces calling functions of the data monitoring device. Vaidya's disclosure of application programming interfaces and other requirements of claimed language is discussed above. Appellant further recites part of rejection to claim 11 (Column 6 lines 1-11) with emphasis, and argues that Vaidya does not disclose use of Application Programming Interfaces. However, column 6 lines 1-11 clearly shows the exchange of data between different modules of Vaidya's system. As mentioned before, this inherently requires the modules to have the capability to initiate an action on a separate device, which is the essence of Application Programming Interfaces. In addition, further details of rejection to claim 11, as described in section

9.4.11 above, clearly indicates how use of Application Programming Interfaces is disclosed by Vaidya alone, or in combination of Vaidya and Porras.

Group #3: Claim 16

Appellant argues that Vaidya does not disclose a technique "wherein the application program interfaces provide parsing of signatures used in signature matching". However, the mention excerpt of Vaidya (Column 10, lines 17-450) clearly shows an attack signature parsed. Therefore, parsing attack signatures are clearly disclosed by Vaidya. It is also established that application program interfaces are generally used to invoke functionalities, and this concept was well-known at the time of invention and also disclosed by Vaidya. Therefore, use of application programming interfaces to parse the signatures is disclosed by Vaidya.

For the above reasons, it is believed that the rejections should be sustained.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

Respectfully submitted,

Farid Homayounmehr

June 15, 2006

Conferees:

Gilberto Barron

Benjamin Lanier

GILBERTO BARRÓN JR
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100